news-please

A Generic News Crawler and Extractor

Felix Hamborg, Norman Meuschke,
Corinna Breitinger, Bela Gipp

Department of Computer and Information Science
University of Konstanz, Germany
[firstname.lastname]@uni-konstanz.de

Abstract

The amount of news published and read online has increased tremendously in recent years, making news data an interesting resource for many research disciplines, such as the social sciences and linguistics. However, large scale collection of news data is cumbersome due to a lack of generic tools for crawling and extracting such data. We present news-please, a generic, multi-language, open-source crawler and extractor for news that works out-of-the-box for a large variety of news websites. Our system allows crawling arbitrary news websites and extracting the major elements of news articles on those websites, i.e., title, lead paragraph, main content, publication date, author, and main image. Compared to existing tools, news-please features full website extraction requiring only the root URL.

Keywords: news crawler; news extractor; scraper; information extraction

1 Introduction and background

News articles are an interesting data source for researchers in many domains. For instance, social scientists heavily rely on news for performing framing analyses, i.e. studying how people interpret situations and activities. News

In: M. Gäde/V. Trkulja/V. Petras (Eds.): Everything Changes, Everything Stays the Same? Understanding Information Spaces. Proceedings of the 15th International Symposium of Information Science (ISI 2017), Berlin, 13th—15th March 2017. Glückstadt: Verlag Werner Hülsbusch, pp. 218—223.

are a well-suited data source for this kind of analysis, since they reflect which events received public attention and how these events were portrayed. Some news data sets, such as RCV1 (Lewis et al., 2004), are freely available. However, researchers often need to compile their own dataset, e.g., to include news published by specific outlets or in a certain time frame. Due to the lack of a publicly available, integrated crawler and extractor for news, researchers often implement such tools redundantly. The process of gathering news data typically consists of two phases: (1) *crawling news websites* and (2) *extracting information from news articles*.

Crawling news websites can be achieved using many web crawling frameworks, such as *scrapy* for Python (Kouzis-Loukas, 2016). Such frameworks traverse the links of websites, hence need to be tailored to the specific use case.

Extracting information from news articles is required to convert the raw data that the crawler retrieves into a format that is suitable for further analysis tasks, such as natural language processing. Information to be extracted typically includes the headline, authors, and main text. *Website-specific extractors*, such as used in (Meschenmoser et al., 2016; Paliouras et al., 2008), must be tailored to the individual websites of interest. These systems typically achieve high precision and recall for their extraction task, but require significant initial setup effort in order to customize the extractors to a set of specific news websites. Such website-specific extractors are most suitable when high data quality is essential, but the number of different websites to process is low.

Generic extractors are intended to obtain information from different websites without the need for adaption. They use heuristics, such as link density and word count, to identify the information to be extracted. Our literature review and experiments have shown that *Newspaper* (Ou-Yang, 2013) is currently one of the most sophisticated and best performing news extractors. It features robust extraction of all major news article elements and supports more than ten languages. Newspaper includes basic crawling, but lacks full website extraction, auto-extraction of new articles, and news content verification, i.e. determining whether a page contains a news article. The extraction performance of other frameworks, such as *boilerpipe* (Kohlschütter, Fankhauser & Nejdl, 2010), *Goose* (Labs, 2016), and *readability* (Baburov, 2010) is lower than that of the *Newspaper* tool. Furthermore, these latter tools do not offer support for crawling websites.

To our knowledge, no available tool fully covers both the crawling and extraction phase for news data. Web crawler frameworks require use-case specific adaptions. News extractors lack comprehensive crawling functionality. Existing systems lack several key features, particularly the capability (1) to extract information from *all* articles published by a news outlet (*full website extraction*) and (2) to auto-extract newly published articles. With *news-please*, we provide a system that addresses these two weaknesses using a generic crawling and extraction approach. The following section presents news-please in detail.

2 System overview

news-please is an open-source news crawler and extractor written in Python developed to meet five requirements: (1) broad coverage – extract news from any outlet's website, (2) full website extraction, (3), high quality of extracted information, (4) ease of use – simple initial configuration, and (5) maintainability. Where possible, news-please uses existing state-of-the-art tools, which we extended with functionality to meet the outlined requirements. This section describes the processing pipeline of news-please as shown in figure 1.

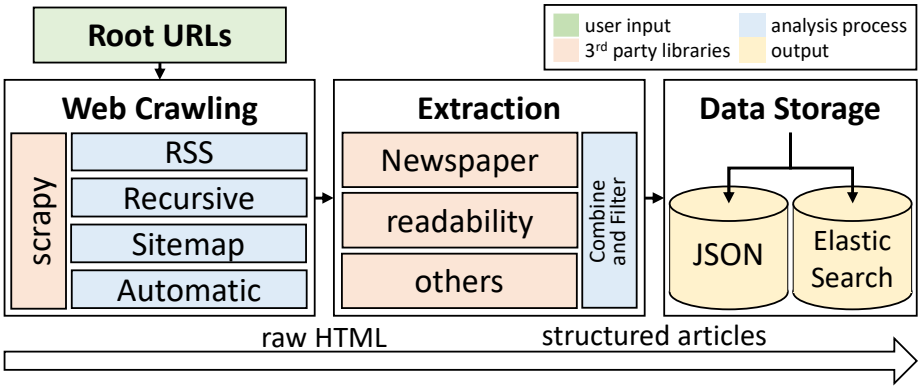

Fig. 1 Pipeline for news crawling and extraction

Root URLs. Users provide URLs that point to the *root* of news outlets' websites, e.g., nyt.com. For each root URL, the following tasks are performed.

Web Crawling. news-please performs two sub-tasks in this phase. (1) The *crawler* downloads articles' HTML, using the *scrapy* framework. (2) To find all articles published by the news outlet, the system supports four techniques: *RSS* (analyzing RSS feeds for recent articles), *recursive* (following internal links in crawled pages), *sitemap* (analyzing sitemaps for links to all articles), and *automatic* (tries sitemaps, falls back to recursive in the case of an error). The approaches can also be combined, e.g., by starting two news-please instances in parallel, one in automatic mode to get all articles published so far, and another instance in RSS mode to retrieve recent articles.

Extraction. We use existing state-of-the-art news extractors to obtain the desired information, i.e., title, lead paragraph, main content, author, date, and main image. In preliminary tests (cf. sect. 3), we evaluated the performance of four extractors (*boilerpipe*, *Goose*, *Newspaper*, and *readability*). *Newspaper* performed best for all news elements combined followed by *readability*. Thus, we integrated both extractors into *news-please*. Because both *Newspaper* and *readability* performed poorly for extracting publication dates, we added a regex-based date extractor (Geva, 2016). Our component-based design allows easily adding or removing extractors in the future. Currently, news-please combines the results of the extractors using rule-based heuristics. We discard pages that are likely not articles using heuristics, such as link-to-headline ratio, and metadata filters.

Data Storage. news-please currently supports writing the extracted data to JSON files and to an Elasticsearch interface.

3 Extraction performance

In tests, we compared the extraction performance of *news-please* against the four extractors: *boilerpipe*, *Goose*, *Newspaper*, and *readability*. In total, we selected 20 articles from 20 news websites (the top 15 news outlets by global circulation and five major outlets in Germany) and manually assessed the quality of extracted information using a four-point-scale: (A) perfect; (B) good: the beginning of an element is extracted correctly, later information is

missing or information from other elements is added, (C) poor: in addition to (B), the beginning of an element is not extracted entirely correctly, (D) unusable: much information is missing or from other elements. news-please performed particularly well for titles (82% in category A or B), description (76% in A or B), date (70% in A or B), and main image (76% in A or B). For other elements, the extraction quality can still be improved: main content (62% in A or B) and author (34% in A or B). Overall, news-please performed better than the included extractors individually.

4 Conclusion and future work

We present *news-please*, the first integrated crawler and information extractor specifically designed for news articles. news-please is able to crawl *all* articles of a news outlet including articles published during the crawling process. The system combines the results of three state-of-the-art extractors. For high maintainability and extendibility, news-please allows inclusion of additional extractors and adaption to use-case-specific requirements, e.g., by adding a SQL result writer. In tests, we found that news-please achieves a higher extraction quality than the individual extractors. By integrating both the crawling and extraction task, researchers can gather news faster and with less initial effort and long-term effort. In the future, we will focus on improving the extraction of an article's main content, supporting more languages, improving the combination of extracted content elements, and evaluating the performance of news-please in more detail. The code has been made available under an Apache 2 license at: https://github.com/fhamborg/news-please.

Acknowledgements

This work has been supported by the Carl Zeiss Foundation. We also thank the students at the University of Konstanz who helped realize news-please: M. Bock, M. Fried, J. Hassler, M. Klatt, K. Kress, S. Lachnit, M. Pafla, F. Schlor, M. Sharinghousen, C. Spener, and M. Steinmaier.

References

Baburov, Y. (2010): python-readability. https://github.com/buriy/python-readability

Geva, R. (2016): article-date-extractor. https://github.com/Webhose/article-date-extractor

Kohlschütter, C., P. Fankhauser, and W. Nejdl (2010): Boilerplate detection using shallow text features. In: *Proceedings of the third ACM international conference on Web search and data mining* (pp. 441–450). ACM.

Kouzis-Loukas, D. (2016): *Learning Scrapy*. Packt Publishing Ltd.

Labs, G. (2016): Goose – Article Extractor. https://github.com/GravityLabs/goose

Lewis, D. D., Y. Yang, T. G. Rose, and F. Li (2004): Rcv1: A new benchmark collection for text categorization research. In: *Journal of machine learning research*, 5 (Apr), 361–397.

Meschenmoser, P., N. Meuschke, M. Hotz, B. Gipp (2016): Scraping Scientific Web Repositories: Challenges and Solutions for Automated Content Extraction. In: *D-Lib Magazine*, 22 (9/10).

Ou-Yang, L. (2013): Newspaper: Article scraping & curation. http://newspaper.readthedocs.io/en/latest/

Paliouras, G., A. Mouzakidis, V. Moustakas, C. Skourlas, C. (2008): PNS: A personalized news aggregator on the web. In: *Intelligent interactive systems in knowledge-based environments* (pp. 175–197). Springer.